Assignment 2 - BINF6210

Jesse Wolf 0830233

2022-10-28

```
# Setting seed for the entire document to ensure reproducible analyses
knitr::opts_chunk$set(warning=FALSE,message=FALSE,
                      fig.width=12, fig.height=8)

set.seed(44)
```

```
# Loading relevant packages and combining invisible with lapply
# to not print boolean statement of TRUE for each package being loaded
libs <- c("tidyverse", "stringi", "ape", "RSQLite",
          "Biostrings", "muscle", "DECIPHER", "phytools",
          "seqinr", "rentrez", "ggmsa", 'patchwork',
          'phangorn','ggtree')

lapply(libs, require, character.only = TRUE)
```

Introduction

Using molecular tools, phylogenies can be easily constructed to represent a diversity of taxonomic groups and organisms. Multiple approaches exist; global alignment looks at the entire length of a given sequence and works especially well for closely related species, while local alignment compares subsets of a given sequence to find similar (or identical) regions (Suárez-Díaz & Muñoz, 2008). Using these approaches, we can identify clades that may be undergoing rapid diversification or extinction and assess the contribution of a given lineage to a group's overall genetic diversity (Moritz, 1995). Members of the *Sciuridae* family exhibit a widespread geographic distribution as well as high species and morphological diversity, thus, making them a suitable model for evolutionary studies (Menéndez et al., 2020). Various markers have been utilized to perform taxonomic classification on the *Sciuridae* family and provide revisions to past molecular phylogenetic analyses (e.g., Balakirev & Rozhnov, 2019; Li et al., 2020; Jackson et al., 2022). As such, the objective of this project was to utilize two different genes, one mitochondrial (Cytochrome c oxidase subunit I; COI) and one nuclear (Interphotoreceptor Retinoid-Binding Protein; IRBP) to build a *Sciuridae* phylogeny using data obtained from the National Center for Biotechnology Information. IRBP is commonly used in mammalian phylogenetic studies and has been utilized in multiple studies investigating *Sciuridae* phylogenetic reconstruction (Li et al., 2020, Jackson et al., 2022). COI is widely used in molecular phylogenetics and has also been applied to conduct molecular phylogenetic analyses within the *Sciuridae* family (Gabrielli et al., 2014).

Code Section 1 - Data Acquisition, Exploration, Filtering, and Quality Control

```
# # Entrez function provided via Courselink FetchFastaFiles
# <- function(searchTerm, seqsPerFile = 10000,
# fastaFileName) { # This function will fetch FASTA files
```

```
# from NCBI nuccore based on a provided search term.  #
# searchTerm = character vector containing Entrez search
# term # seqsPerFile = number of sequences to write to each
# FASTA file # fastaFileName = character vector containing
# name you want to give to the FASTA files you are fetching
# #Initial search for finding maximum number of hits
# search1 <- entrez_search(db = 'nuccore', term =
# searchTerm) # Second search for obtaining max number of
# hits and their IDs search2 <- entrez_search(db =
# 'nuccore', term = searchTerm, retmax = search1$count,
# use_history = T) # Fetch the sequences in FASTA format
# using the web_history object.  for (start_rec in seq(0,
# search2$retmax, seqsPerFile)) { fname <-
# paste(fastaFileName, start_rec, '.fasta', sep = '') recs
# <- entrez_fetch(db = 'nuccore', web_history =
# search2$web_history, rettype = 'fasta', retstart =
# start_rec, retmax = seqsPerFile) write(recs, fname)
# print(paste('Wrote records to ', fname, sep = '')) }
# return(search2) }


# Using the above function to retrieve Sciuridae data from
# NCBI from both the COI and IRBP genes

# COI is typically between 400 and 7000 base pairs long (Lv
# et al. 2014), while IRBP is typically around 1000 bp
# long, but there is more variation relative to COI (Li et
# al. 2020).

# The below lines are commented out to avoid re-running
# them each time the script is executed.

# COI_search<- FetchFastaFiles(searchTerm =
# 'Sciuridae[ORGN] AND COI[Gene] AND 400:700[SLEN]',
# fastaFileName = 'sciuridaetest_COI.fa') IRBP_search <-
# FetchFastaFiles(searchTerm = 'Sciuridae[ORGN] AND
# IRBP[Gene] AND 500:1300[SLEN]', fastaFileName =
# 'sciuridaetest_IRBP.fa')

# Reading in the exported COI Sciuridae fasta as a string
# set for downstream filtering and analysis
Sciuridae_stringSet_COI <- readDNAStringSet("sciuridaetest_COI.fa0.fasta")

# Convert to dataframe format, adding a column named COI
# title and COI sequence to produce a cleaner to work with
# data frame
dfSciuridae_COI <- data.frame(COI_Title = names(Sciuridae_stringSet_COI),
    COI_Sequence = paste(Sciuridae_stringSet_COI))

# Using the word function to extract the second and third
# word from the COI_title column and creating a new column
# with this information called Species_Name
dfSciuridae_COI$Species_Name <- word(dfSciuridae_COI$COI_Title,
```

```
    2L, 3L)

# Reorganize the columns to start with COI title, species
# name, and the COI sequence
dfSciuridae_COI <- dfSciuridae_COI[, c("COI_Title", "Species_Name",
    "COI_Sequence")]

# Taking a look at our cleaned up data frame
dim(dfSciuridae_COI)
str(dfSciuridae_COI)
class(dfSciuridae_COI)

# Reading in the exported IRBP Sciuridae fasta as a string
# set for downstream filtering and analysis
Sciuridae_stringSet_IRBP <- readDNAStringSet("sciuridaetest_IRBP.fa0.fasta")

# Convert to dataframe format, adding a column named IRBP
# title and IRBP sequence to produce a cleaner to work with
# data frame
dfSciuridae_IRBP <- data.frame(IRBP_Title = names(Sciuridae_stringSet_IRBP),
    IRBP_Sequence = paste(Sciuridae_stringSet_IRBP))
str_count(dfSciuridae_IRBP$IRBP_Sequence)

# Using the word function to extract the second and third
# word from the IRBP_title column and creating a new column
# with this information called Species_Name
dfSciuridae_IRBP$Species_Name <- word(dfSciuridae_IRBP$IRBP_Title,
    2L, 3L)

# Reorganize the columns to start with COI title, species
# name, and the IRBP sequence
dfSciuridae_IRBP <- dfSciuridae_IRBP[, c("IRBP_Title", "Species_Name",
    "IRBP_Sequence")]

# Taking a look at our cleaned up data frame
dim(dfSciuridae_IRBP)
str(dfSciuridae_IRBP)
class(dfSciuridae_IRBP)

# Checking how many unique species we have from the IRBP
# dataset and how many rows of data we have
length(unique(dfSciuridae_IRBP$Species_Name))
length(dfSciuridae_IRBP$Species_Name)

# We have 83 unique species with IRBP data from NCBI, but
# there are 154 rows of data

# Let's sample 1 individual per species from the IRBP
# dataset, grouping by species name and sampling n=1 from
# each species.
dfSciuridae_IRBP_Subset <- dfSciuridae_IRBP %>%
    group_by(Species_Name) %>%
    sample_n(1)
```

```
# Confirming our sub-sampling worked by using the all.equal
# function to test if the length of unique values from our
# non-sub-sampled IRBP data frame is equal to the number of
# rows in our sub-sampled data frame
all.equal(length(unique(dfSciuridae_IRBP$Species_Name)), nrow(dfSciuridae_IRBP_Subset))

# Let's do the same for the COI gene

# Checking how many unique species we have from the COI
# dataset and how many rows of data we have
length(unique(dfSciuridae_COI$Species_Name))
length(dfSciuridae_COI$Species_Name)

# We have 69 unique species from the COI dataset, but there
# are 475 rows of data

# Let's sample 1 individual per species from the COI
# dataset, grouping by species name and sampling n=1 from
# each species
dfSciuridae_COI_Subset <- dfSciuridae_COI %>%
    group_by(Species_Name) %>%
    sample_n(1)

# Confirming our sub-sampling worked by using the all.equal
# function to test if the length of unique values from our
# non-sub-sampled COI data frame is equal to the number of
# rows in our sub-sampled data frame
all.equal(length(unique(dfSciuridae_COI$Species_Name)), nrow(dfSciuridae_COI_Subset))

# Creating a new data frame that combines COI and IRBP
# Sciuridae data by species name, meaning whenever species
# names are the same, the sequence data will be appended so
# we can compare them in downstream analyses. The all =
# FALSE parameter ensures that we do not retain sequence
# data when there is only data for either COI/IRBP. As we
# are planning to do a comparative analysis using the two
# markers, it would be not helpful to have some species
# with data for only one of the markers and not both.
dfSciuridae_AllSeqs <- merge(dfSciuridae_IRBP_Subset, dfSciuridae_COI_Subset,
    by = "Species_Name", all = F)
dim(dfSciuridae_AllSeqs)
str(dfSciuridae_AllSeqs)
class(dfSciuridae_AllSeqs)

# Checking out our combined data frame - making sure there
# are no NA's in our sequence data
names(dfSciuridae_AllSeqs)
sum(is.na(dfSciuridae_AllSeqs$IRBP_Sequence))
sum(is.na(dfSciuridae_AllSeqs$COI_Sequence))

# Checking the summary of our sequence length data to make
# sure there are no outliers and that our SLEN parameter in
# our orginal search was successful
```

```
summary(str_length(dfSciuridae_AllSeqs$IRBP_Sequence))
summary(str_length(dfSciuridae_AllSeqs$COI_Sequence))

# Checking the average amount of A's and T's within both
# datasets, to get a general idea of the relative
# nucleotide composition. In both instances, the average
# amount of A's and T's are greater in the IRBP sequence
# data, which is likely a product of the longer sequence
# length in general.
mean(str_count(dfSciuridae_AllSeqs$IRBP_Sequence, "A"))
mean(str_count(dfSciuridae_AllSeqs$COI_Sequence, "A"))
mean(str_count(dfSciuridae_AllSeqs$IRBP_Sequence, "T"))
mean(str_count(dfSciuridae_AllSeqs$COI_Sequence, "T"))

# Creating variables for what we are allowing in terms of
# missing data and length variability - this will allow us
# to filter out sequences with greater than 1% missing data
# and that are >125 nucleotides above or below the median
# sequence length. We decided on these less-stringent
# parameters relative to those seen in lecture due to the
# small size of our dataset.
missing.data <- 0.01
length.var <- 125

# Filtering our data set prior to alignment - we already
# know there are no NA's and that our sequence data only
# contains the two relevant markers that we are studying,
# so filtering in this instance only consists of removing
# leading and trailing N's, as well as all internal gaps
# (represented by -). We also discard any sequences with
# over 1% missing data and any sequences that are 125
# nucleotides above or below the median sequence length.

dfSciuridae_AllSeqs_trim <- dfSciuridae_AllSeqs %>%
    mutate(COI_seq_trim = str_remove_all(COI_Sequence, "^N+|N+$|-")) %>%
    mutate(IRBP_seq_trim = str_remove_all(IRBP_Sequence, "^N+|N+$|-")) %>%
    filter(str_count(COI_seq_trim, "N") <= (missing.data * str_count(COI_seq_trim))) %>%
    filter(str_count(IRBP_seq_trim, "N") <= (missing.data * str_count(IRBP_seq_trim))) %>%
    filter(str_count(COI_seq_trim) >= median(str_count(COI_seq_trim)) -
        length.var & str_count(COI_seq_trim) <= median(str_count(COI_seq_trim)) +
        length.var) %>%
    filter(str_count(IRBP_seq_trim) >= median(str_count(IRBP_seq_trim)) -
        length.var & str_count(IRBP_seq_trim) <= median(str_count(IRBP_seq_trim)) +
        length.var)

# Performing the final check of our combined COI and IRBP
# dataset to ensure it is as clean as possible for
# downstream analyses.

# We can see that the above filtering step removed two
# individuals (Dremomys rufigenis and Tamiops swinhoei).

dim(dfSciuridae_AllSeqs_trim)
```

```
unique(dfSciuridae_AllSeqs_trim$Species_Name)
sum(str_count(dfSciuridae_AllSeqs_trim$COI_seq_trim, "-"))
sum(str_count(dfSciuridae_AllSeqs_trim$IRBP_seq_trim, "-"))
summary(str_count(dfSciuridae_AllSeqs_trim$COI_seq_trim))
summary(str_count(dfSciuridae_AllSeqs_trim$IRBP_seq_trim))

# It looks like we have no internal gaps in our sequences
# for both markers, 25 unique species, and a sequence
# length distribution for each marker that matches what we
# expect.
```

Code Section 1 - Data Acquisition, Exploration, Filtering, and Quality Control - Figures 1 and 2

```
# To be able to align our sequences, we need to change the
# format of the nucleotide data to be suitable for analysis
# with the muscle package, so we need both the COI and IRBP
# columns to be in DNAStringSet format
class(dfSciuridae_AllSeqs_trim)
dfSciuridae_AllSeqs_trim$COI_seq_trim <- DNAStringSet(dfSciuridae_AllSeqs_trim$COI_seq_trim)
dfSciuridae_AllSeqs_trim$IRBP_seq_trim <- DNAStringSet(dfSciuridae_AllSeqs_trim$IRBP_seq_trim)

# Confirming the above worked
class(dfSciuridae_AllSeqs_trim$COI_seq_trim)
class(dfSciuridae_AllSeqs_trim$IRBP_seq_trim)

# Using the word function to extract the first word from
# the IRBP/COI title, so that we are sure to carry forward
# a unique identifier for the nucleotides through the
# analysis steps.
names(dfSciuridae_AllSeqs_trim$IRBP_seq_trim) <- word(dfSciuridae_AllSeqs_trim$Species_Name,
    1L, 2L)
names(dfSciuridae_AllSeqs_trim$IRBP_seq_trim)

names(dfSciuridae_AllSeqs_trim$COI_seq_trim) <- word(dfSciuridae_AllSeqs_trim$Species_Name,
    1L, 2L)
names(dfSciuridae_AllSeqs_trim$COI_seq_trim)

# Taking a look at the sequences from both markers on
# screen and in a browser for data quality control
dfSciuridae_AllSeqs_trim$COI_seq_trim
dfSciuridae_AllSeqs_trim$IRBP_seq_trim

BrowseSeqs(dfSciuridae_AllSeqs_trim$COI_seq_trim)
BrowseSeqs(dfSciuridae_AllSeqs_trim$IRBP_seq_trim)

# There seems to be more sequence length variability in the
# IRBP gene in general, but hopefully our filtering step
# was strict enough to allow for a proper alignment.

# Using the muscle algorithm from the muscle package, let's
```

```r
# perform a multiple sequence alignment.

# Running muscle on our IRBP sequences, converting the
# alignment to a DNAStringSet, creating a log file with
# verbose set to TRUE so that we can refer to the the
# parameters that were run for each alignment.
sciuridae_IRBP_alignment <- DNAStringSet(muscle::muscle(dfSciuridae_AllSeqs_trim$IRBP_seq_trim,
    log = "IRBP_log.txt", verbose = T), use.names = T)

# Running muscle on our COI sequences, converting the
# alignment to a DNAStringSet, creating a log file with
# verbose set to TRUE so that we can refer to the the
# parameters that were run for each alignment.
sciuridae_COI_alignment <- DNAStringSet(muscle::muscle(dfSciuridae_AllSeqs_trim$COI_seq_trim,
    log = "COI_log.txt", verbose = T), use.names = T)

# Viewing both alignments in a browser.
BrowseSeqs(sciuridae_IRBP_alignment)
BrowseSeqs(sciuridae_COI_alignment)

# Let's have a look at the sequence length of our
# alignments between genes - it looks like the IRBP
# alignment is much larger (1289 nucleotides versus 726 in
# the consensus sequences).
length(sciuridae_IRBP_alignment[[1]])
length(sciuridae_COI_alignment[[1]])

# Let's also see how many gaps each alignment has, on
# average.

# The mean number of gaps in the IRBP alignment is 104.84,
# relative to 74.92 in COI.
sciuridae_COI_alignment %>%
    lapply(str_count, "-") %>%
    unlist %>%
    mean

sciuridae_IRBP_alignment %>%
    lapply(str_count, "-") %>%
    unlist %>%
    mean

# We can see here that the IRBP alignment has slightly
# longer gaps on average. We will try to perform alignments
# with different parameters/modifications to improve the
# IRBP alignment.

# Seeing as the IRBP alignment has many large gaps and
# looks visually problematic, let's try increasing the gap
# penalty and the number of iterations.
sciuridae_IRBP_alignment_largegap <- DNAStringSet(muscle::muscle(dfSciuridae_AllSeqs_trim$IRBP_seq_trim
    gapopen = -10000, maxiter = 16, log = "COI_log_largegap.txt",
    verbose = T), use.names = T)
```

```
# Viewing alignment
BrowseSeqs(sciuridae_IRBP_alignment_largegap)

# Now, checking length of the alignment with the heavier
# gap penalty - we can see that there is no difference when
# using a larger gap penalty, but it was a useful check to
# see if we could improve the alignment!
length(sciuridae_IRBP_alignment_largegap[[1]])

sciuridae_IRBP_alignment_largegap %>%
    lapply(str_count, "-") %>%
    unlist %>%
    mean

# Just so we are sure that we are using the best alignment
# possible, let's try a few different packages and see how
# they perform.

# Using the DECIPHER package to perform alignments and
# setting refinements = 2 to realign groups of sequences
# back onto the original alignment and then having the
# function output the best of the two alignments.
decipher_alignment_COI <- AlignSeqs(dfSciuridae_AllSeqs_trim$COI_seq_trim,
    refinements = 2)
# Viewing alignment
BrowseSeqs(decipher_alignment_COI)

decipher_alignment_IRBP <- AlignSeqs(dfSciuridae_AllSeqs_trim$IRBP_seq_trim,
    refinements = 2)
# Viewing alignment
BrowseSeqs(decipher_alignment_IRBP)

# The alignment looks the same for both COI and IRBP -
# great!

# These alignments look the same as those output by muscle
# but let's confirm quantitatively.

# The average number of gaps for both alignments are
# identical, so we will stick with the muscle alignments
# for downstream analyses.
decipher_alignment_COI %>%
    lapply(str_count, "-") %>%
    unlist %>%
    mean

decipher_alignment_IRBP %>%
    lapply(str_count, "-") %>%
    unlist %>%
    mean

# Let's make a histogram of the amount of gaps to look at
# the final alignments.  We are using base R to generate
```

```
# this plot, labelling both axes as well as including a
# title. We manually set the limits of our x-axis, specify
# to use 25 breaks within the data, and use the lapply
# function to count all gaps within a given alignment. We
# use the add = T parameter in the second plot to add it to
# the first plot and make use of the unlist function to
# produce a vector which contains all the atomic components
# which occur in our list of gaps.
hist(unlist(lapply(sciuridae_COI_alignment, str_count, "-")),
    breaks = 25, xlim = c(0, 250), col = rgb(1, 0, 0, 0.5), xlab = "Size of gaps",
    ylab = "Frequency", main = "Distribution of size of gaps in sequence alignments")
hist(unlist(lapply(sciuridae_IRBP_alignment, str_count, "-")),
    breaks = 25, xlim = c(0, 250), col = rgb(0, 0, 1, 0.5), add = T)
legend("topright", legend = c("COI", "IRBP"), col = c(rgb(1,
    0, 0, 0.5), rgb(0, 0, 1, 0.5)), pt.cex = 2, pch = 15)
```

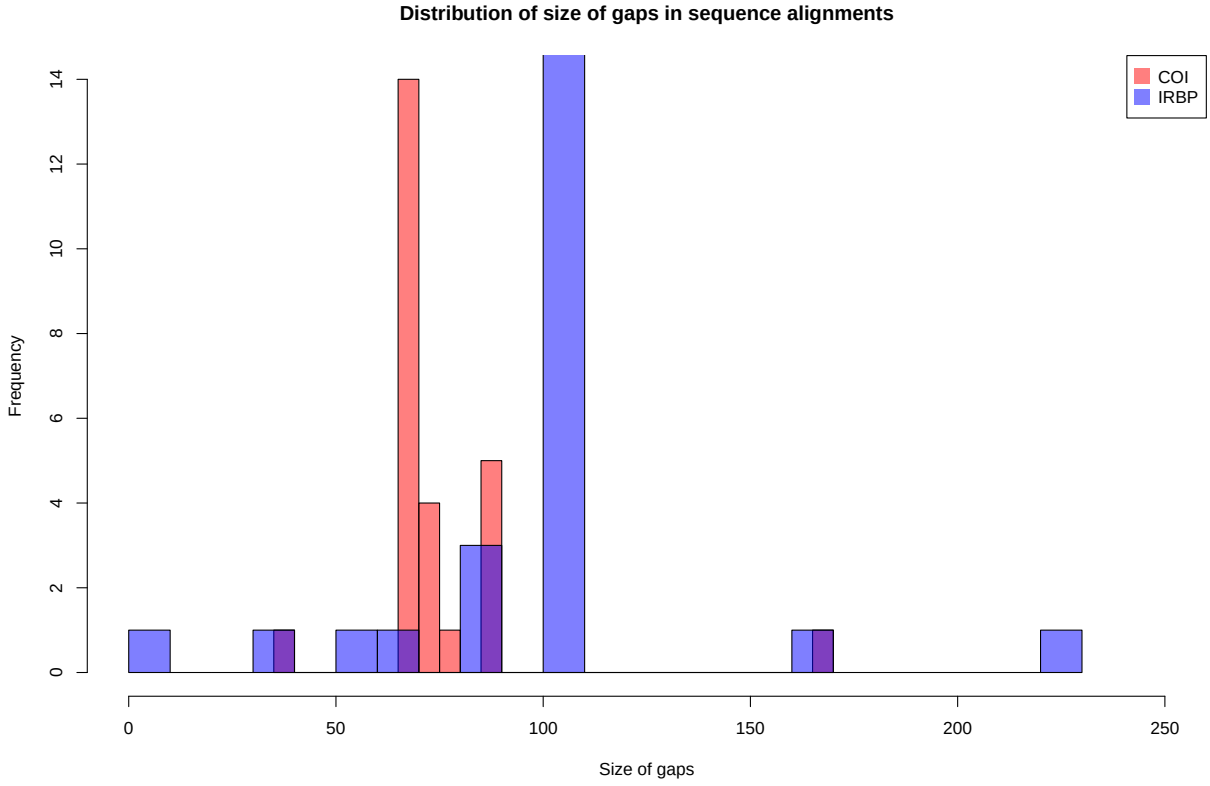

Figure 1: A histogram of data from both the COI and IRBP gene depicting the frequency of gaps of various sizes present within both alignments.

```
# Using the ggmsa package, we can plot a subset of our
# multiple sequence alignment to visually determine sites
# where there is a clear consensus sequence (an invariant
# site) relative to more variant sites. For example, we can
# see at the beginning of our IRBP alignment (250 bp), all
# but one individual have a T, indicating a completely
# invariant site. Conversely, both at the 284th and 298th
# base pair in our IRBP alignment, it is highly variable,
```

```
# indicated by the stacked set of nucleotides and a
# difference of colour between individuals.

ggmsa(sciuridae_IRBP_alignment, start = 250, end = 300, font = NULL,
    color = "Chemistry_NT") + geom_seqlogo(color = "Chemistry_NT") +
    geom_msaBar()
```

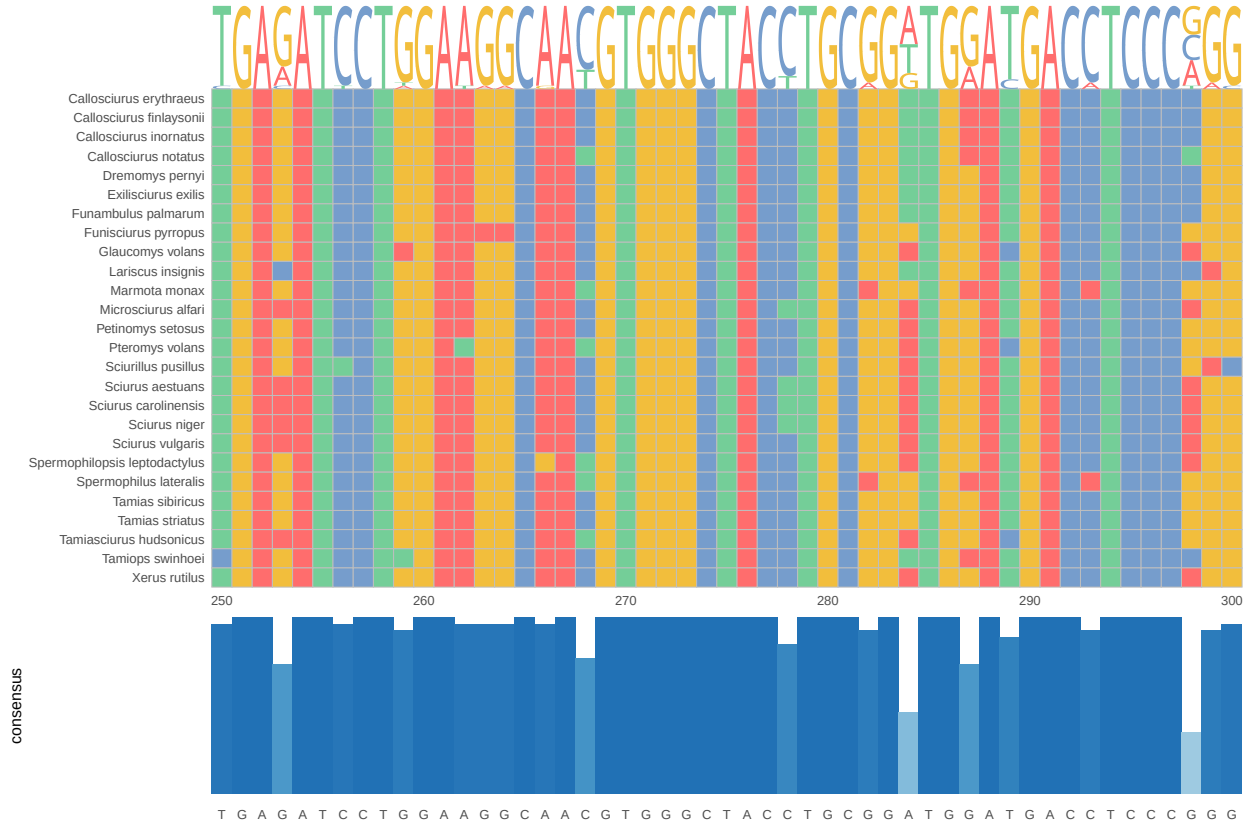

Figure 2: A ggplot of our multiple sequence alignments for the IRBP gene. Species are on the left and the alignment for each species from the 250th to 300th nucleotide is displayed (n=25). Above the individual alignments is the consensus sequence; nucleotide positions where only one letter is present indicates an invariant site, where all individuals have the same nucleotide, where more than one nucleotide letter is present indicates a variant site, indicating a difference of nucleotides between species at a given position. Below the alignment is a histogram depicting the relative frequency of the consensus nucleotide. A 100% consensus nucleotide would be represented by the longest bar and the darkest shade of blue. A <100% consensus nucleotide would be relatively shorter and lighter in colour.

Code Section 2 - Main Analysis - Figure 3

```
# Writing both alignments to file - commented out to avoid
# writing a new fasta every time the script is run
# writeXStringSet(sciuridae_COI_alignment,
# 'sciuridae_COI_alignment.fas', format = 'fasta')
# writeXStringSet(sciuridae_IRBP_alignment,
# 'sciuridae_IRBP_alignment.fas', format = 'fasta')
```

```
# Reading back in our fasta files and using the read.dna
# function to turn them into matrices for distance
# calculations downstream
COI_alignment <- read.dna("sciuridae_COI_alignment.fas", format = "fasta",
    as.matrix = TRUE)
IRBP_alignment <- read.dna("sciuridae_IRBP_alignment.fas", format = "fasta",
    as.matrix = TRUE)

# Creating distance matrices for each gene, using the
# Kimura 2-parameter equal base frequencies and accounts
# for the difference between transitions and transversions
# with one parameter
COI_distanceMatrix <- dist.dna(COI_alignment, model = "K80",
    as.matrix = TRUE, pairwise.deletion = TRUE)
IRBP_distanceMatrix <- dist.dna(IRBP_alignment, model = "K80",
    as.matrix = TRUE, pairwise.deletion = TRUE)

# Taking a look at our COI/IRBP distance matrices
head(COI_distanceMatrix)
head(IRBP_distanceMatrix)

# Plotting preliminary Unweighted Pair Group Method with
# Arithmetic Mean trees for each gene.

# UPGMA is a cluster distance-based method is a preliminary
# approach, as it is not very robust to variability in
# rates of molecular evolution and as we are dealing with a
# diverse family of organisms, this approach likely does
# not represent the true phylogeny.
COI_upgma <- upgma(COI_distanceMatrix)
IRBP_upgma <- upgma(IRBP_distanceMatrix)

# Generating a preliminary Neighbour-Joining tree
# estimation for both genes using distance matrices.

# Neighbour joining is an improvement relative to UPGMA,
# however, the accuracy is generally not as good as
# explicit model based methods, such as maximum likelihood.
COI_nj <- NJ(COI_distanceMatrix)
IRBP_nj <- NJ(IRBP_distanceMatrix)

# We can use the parsimony function to compare the
# parsimony score between UPGMA and NJ trees for both
# markers. The parsimony score is the number of changes
# which are at least necessary to describe the data for a
# given tree.

# First, we need to convert our alignment to a .phydat
# object to use the parsimony function. It will also be
# used by modeltest.
COI_alignment.phydat <- as.phyDat(COI_alignment)
IRBP_alignment.phydat <- as.phyDat(IRBP_alignment)
```

```
# Parsimony scores for our COI data indicate that there are
# slightly less changes that are necessary to describe the
# data for our neighbour-joining tree, however, the scores
# are quite close (1668 for UPGMA versus 1647 for Neighbour
# Joining).
parsimony(COI_upgma, COI_alignment.phydat)
parsimony(COI_nj, COI_alignment.phydat)

# Parsimony scores for our IRBP data indicate that there
# are slightly less changes that are necessary to describe
# the data for our neighbour-joining tree, however, the
# scores are quite close (608 for UPGMA versus 604 for
# Neighbour Joining).
parsimony(IRBP_upgma, IRBP_alignment.phydat)
parsimony(IRBP_nj, IRBP_alignment.phydat)

# Ultimately we will use maximum likelihood approaches to
# generate phylogenies for both the COI and IRBP genes for
# the Sciuridae family.

# The model used to calculate pairwise distance by
# Balakirev & Rozhnov, 2019; Li et al., 2020; and Jackson
# et al., 2022 was the general time-reversible model with
# gamma distribution and invariable cites (GTR+G+I). Here I
# will perform some model testing to determine the best
# nucleotide substitution model for our data.

# Working with the COI data to generate a maximum
# likelihood tree

# Compare different nucleotide substitution models to
# determine which model is best moving forward. We use the
# Neighbour joining tree here as it was slightly more
# parsimonious relative to our UPGMA tree. We test a
# handful of common models and include the gamma
# distribution and invariable sites parameters as past
# research on Sciuridae phylogenetics utilized them. K=4 is
# the default number of rate classes.
COI_alignment.modeltest <- modelTest(COI_alignment.phydat, tree = COI_nj,
    model = c("JC", "F81", "K80", "HKY", "SYM", "GTR"), G = TRUE,
    I = TRUE, k = 4)

# Using the model test above, we can check the model with
# the lowest Aikake's Information Criterion (AIC), which
# indicates best model fit.
COI_alignment.modeltest$Model[COI_alignment.modeltest$AIC ==
    min(COI_alignment.modeltest$AIC)]

# It turns out that the best model is the
# Hasegawa-Kishino-Yano (HKY+G(4)) model and we can use the
# pml function to compute the likelihood of our
# phylogenetic tree. The k parameter here tells pml to use
# 4 intervals of the discrete gamma distribution.
```

```
COI_alignment.pml <- pml(COI_nj, COI_alignment.phydat, model = "HKY",
    k = 4)

# Using our initial maximum likelihood alignment, we can
# use the optimization function from the phangorn package
# to optimize the different model parameters. In this case,
# we optimize tree topology, base frequencies, rate
# matrices, proportion of variable size, the gamma rate
# parameter, and edge lengths.
COI_alignment.pml <- optim.pml(COI_alignment.pml, optNni = TRUE,
    optBf = TRUE, optQ = TRUE, optInv = TRUE, optGamma = TRUE,
    optEdge = TRUE)

# In addition, we want to run bootstraps on our maximum
# likelihood alignment and use the bootstrap.pml function
# to do so. We bootstrap 100 samples, return a tree, and
# further optimize tree topology. This is commented out to
# avoid bootstrapping each time the code is run.
# COI_alignment.pml.bs <-
# bootstrap.pml(COI_alignment.pml,bs=100,trees=TRUE,
# optNni=TRUE)

# Here, we plot a tree with bootstrap values included. We
# specify that we want to generate a phylogram and our font
# size to be used. This is commented out as I would only
# like to display the phylogeny with bootstrap values for
# the IRBP alignment.

# plotBS(COI_alignment.pml$tree, COI_alignment.pml.bs,
# type='phylogram', cex=0.5)

# Doing the same as above with the IRBP gene to generate a
# maximum likelihood tree.

# Compare different nucleotide substitution models to
# determine which model is best moving forward. We use the
# Neighbour joining tree here as it was slightly more
# parsimonious relative to our UPGMA tree. We test a
# handful of common models and include the gamma
# distribution and invariable sites parameters as past
# research on Sciuridae phylogenetics utilized them. K=4 is
# the default number of rate classes.
IRBP_alignment.modeltest <- modelTest(IRBP_alignment.phydat,
    tree = IRBP_nj, model = c("JC", "F81", "K80", "HKY", "SYM",
        "GTR"), G = TRUE, I = TRUE, k = 4)

# Using the model test above, we can check the model with
# the lowest Aikake's Information Criterion (AIC), which
# indicates best model fit.
IRBP_alignment.modeltest$Model[IRBP_alignment.modeltest$AIC ==
    min(IRBP_alignment.modeltest$AIC)]

# It turns out that the best model is what was used by the
```

```
# authors mentioned above - GTR+G(4)and we can use the pml
# function to compute the likelihood of our phylogenetic
# tree. The k parameter here tells pml to use 4 intervals
# of the discrete gamma distribution.
IRBP_alignment.pml <- pml(IRBP_nj, IRBP_alignment.phydat, model = "GTR",
    k = 4)

# Using our initial maximum likelihood alignment, we can
# use the optimization function from the phangorn package
# to optimize the different model parameters. In this case,
# we optimize tree topology, base frequencies, rate
# matrices, proportion of variable size, the gamma rate
# parameter, and edge lengths.
IRBP_alignment.pml <- optim.pml(IRBP_alignment.pml, optNni = TRUE,
    optBf = TRUE, optQ = TRUE, optInv = TRUE, optGamma = TRUE,
    optEdge = TRUE)

# In addition, we want to run bootstraps on our maximum
# likelihood alignment and use the bootstrap.pml function
# to do so. We bootstrap 100 samples, return a tree, and
# further optimize tree topology.
IRBP_alignment.pml.bs <- bootstrap.pml(IRBP_alignment.pml, bs = 100,
    trees = TRUE, optNni = TRUE)

# Here, we plot a tree with bootstrap values included. We
# specify that we want to generate a phylogram and our font
# size to be used.
plotBS(IRBP_alignment.pml$tree, IRBP_alignment.pml.bs, type = "phylogram",
    cex = 1)
```

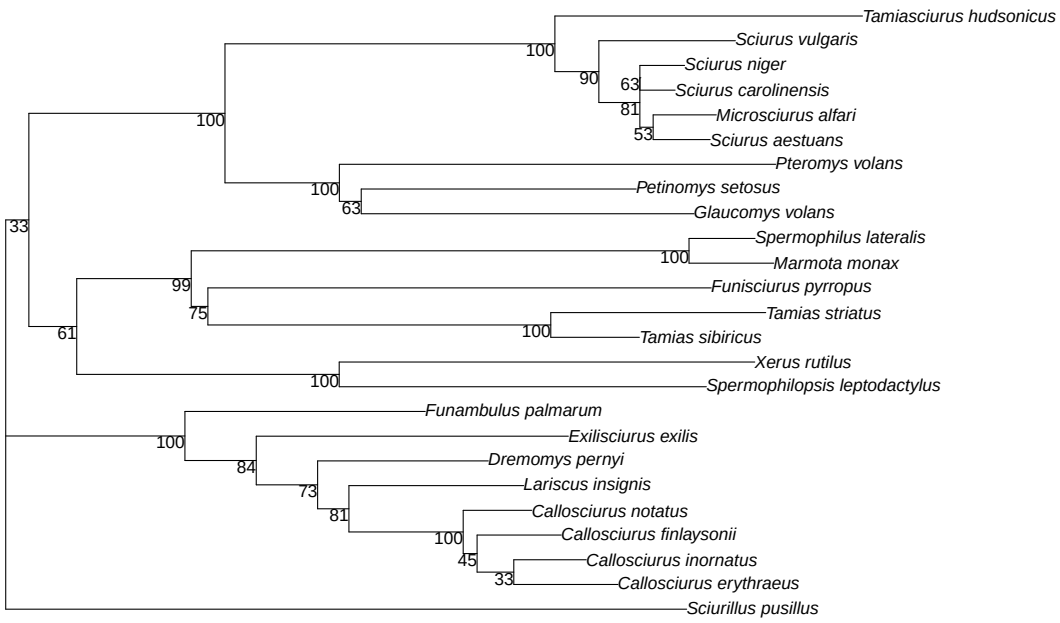

Figure 3: A maximum likelihood tree built using an alignment of the Interphotoreceptor Retinoid-Binding Protein gene from *Sciuridae* individuals (n=25). The Hasegawa-Kishino-Yano model was used as well as four intervals of the discrete gamma distribution. The tree was optimized and 100 bootstrap samples were included. Bootstrap values are included at each node.

Code Section 2 - Main Analysis - Figure 4

```
# Converting our maximum likelihood trees to phylo objects for downstream analyses.
IRBP_phylo <- as.phylo(IRBP_alignment.pml$tree)
COI_phylo <- as.phylo (COI_alignment.pml$tree)

# Quantitative comparison of phylogenies
# We can see that between our COI and IRBP alignments, 10/48 splits are shared
#and 12 clades are not common, indicating our trees are relatively
#different from one another.
comparePhylo(COI_phylo, IRBP_phylo, plot = FALSE, force.rooted = FALSE,
             use.edge.length = FALSE, commons = TRUE,
             legend = "bottomleft")
# We can check the sum of edge lengths between phylogenies. The edge length values
#represent genetic distances in our phylogram. The edge length sums are vastly different
#(0.565 in our IRBP phylogram relative to 20.72 in our COI phylogram).
#This indicates that the amount of genetic distance among species within our
#COI phylogram is much larger when compared to the IRBP phylogram.

sum(COI_phylo$edge.length)
```

```
sum(IRBP_phylo$edge.length)

# Now we know that the trees differ fairly substantially, let's take a look at
#them side by side.
COI_ggtree<- msaplot(ggtree(COI_phylo) +
# Adds labels so each tip has a species name associated with it
  geom_tiplab() +
# Creates points at each node and makes it slightly transparent
    geom_point(alpha = 0.35) +
# Scales our tree to depict relative divergence
  geom_treescale(x = 0, y = -0.5,
                 width = 1)+
# Removes the legend
    theme(legend.position = "none"),
# Uses msaplot and our alignment fasta file to show our alignment alongside
#the phylogeny from 150bp - 175bp
        fasta="sciuridae_COI_alignment.fas", width = 1, offset = 4, window=c(150, 175))

IRBP_ggtree<- msaplot(ggtree(IRBP_phylo, branch.length = "none") +
# Adds labels so each tip has a species name associated with it
  geom_tiplab()+
# Scales our tree to depict relative divergence
    geom_treescale(x = 0, y = -1,
                   width = 25)+
# Creates points at each node and makes it slightly transparent
    geom_point(alpha = 0.35)+
# Moves the legend to the right side
    theme(legend.position = "right"),
# Uses msaplot and our alignment fasta file to show our alignment alongside the phylogeny
#from 150bp - 175bp
        fasta="sciuridae_IRBP_alignment.fas", width = 1, offset = 8.25, window=c(150, 175))
# Combines our two figures into one and tells patchwork to output it in one column
combined_ggtree <- COI_ggtree +
  IRBP_ggtree +
  plot_layout(ncol = 1)
# Prints out final figure to the screen
combined_ggtree
```

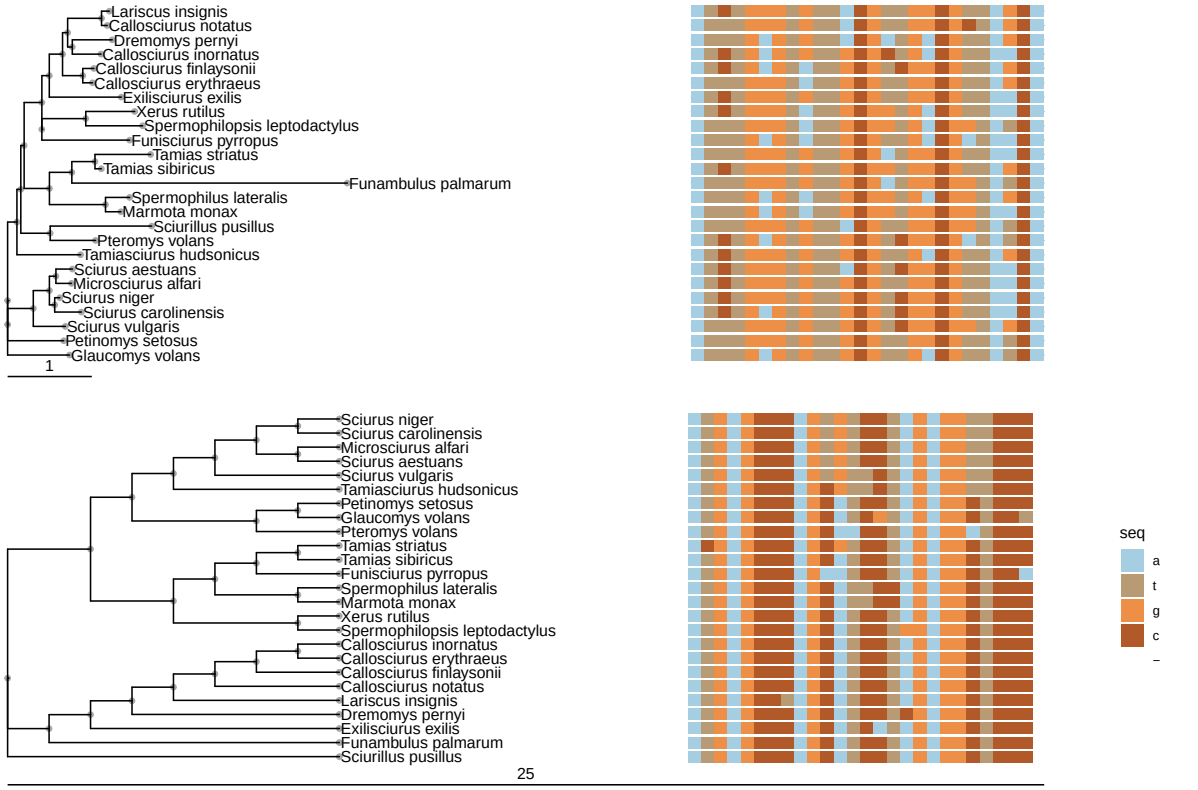

Figure 4: Two phylogenetic trees generated for the family *Sciuridae* along with a subset of the aligned sequences from 150bp to 175bp. The top phylogeny is generated using sequence data from COI gene and the bottom phylogeny is generated using sequence data from the IRBP gene. The aligned sequences on the right are provided as an example of our sequence alignments differ and how that impact the corresponding phylogeny. Colours represent a given nucleotide and the single number above the horizonal line below each individual phylogeny represent genetic distance. As our branch lengths are longer in the IRBP phylogeny, the genetic distance value is also greater. Points in both phylogenies represent individual nodes.

Results and Discussion

As we can see from Figure 3, the bootstrapped values for our IRBP phylogeny were relatively high, indicating that using this gene, we can be more confident that the observed branches are accurate and depict the true phylogeny, given the data provided. Moving forward, this may indicate that nuclear genes are more effective at depicting *Sciuridae* phylogenies. Discrepancies between phylogenies built with mitochondrial markers relative to nuclear markers have been documented (e.g., Shaw, 2002). As such, it has been posited that speciation histories based on mitochondrial genes alone can be misleading. As the mitochondrial genome has a higher mutation rate relative to the nuclear genome, we can expect that the branches of the tree generated with nuclear data should be shorter when compared to the tree generated with mitochondrial data (e.g., Popadin et al., 2022). Longer branches indicate greater genetic change/divergence, which can be expected when using genes with higher mutation rates. However, the data presented here indicate longer branch lengths within the IRBP phylogeny (Figure 4). Furthermore, this could be a product of a difference in molecular evolutionary rates. Ultimately, the results of this study indicate that when building a phylogeny within the *Sciuridae* family, nuclear and mitochondrial genes do not generate congruent phylogenies. In addition, when using a very common mitochondrial gene (COI), bootstrap values were much lower relative to our nuclear gene. This may indicate that COI is not an effective gene to use when attempting to build a phylogeny for *Sciuridae*.

When looking at the *Funambulus palmarum* individual in the COI phylogeny in Figure 4, a large genetic distance is noted relative to the rest of the phylogeny. This could indicate that there was issues with the sequence which was not corrected for by the filtering in this experiment, or, significant divergence relative to the other species. However, this was not observed within the IRBP phylogeny, possibly leading to the conclusion that the divergence seen is not a true representation of the phylogeny, but an issue with the sequence itself. One general caveat to this study was that, rather than generate a consensus sequence from all individuals from the same species that were obtained via NCBI, I decided to randomly subsample one individual. To ensure reproducibility, I used the set.seed argument and applied it globally to the entire workflow. The reason that I chose to subsample a random individual, is because this study is a comparison of the ability of two different genes to depict a phylogeny, not a study that looks to identify the consensus phylogeny. If the goal was to produce a consensus phylogeny for *Sciuridae*, it would be more appropriate to generate a consensus sequence from all available individuals.

Acknowledgements

I would like to thank Linoy Jacobs and Thomas Papp-Simon for their feedback and constructive criticism of the analyses and figures presented here. The FetchFastaFiles function used in this project was obtained directly from Courselink (BINF*6210; https://courselink.uoguelph.ca/d2l/le/content/775979/viewContent/3285915/View).

References Cited

Balakirev, A. E., & Rozhnov, V. V. (2019). Taxonomic revision of beautiful squirrels (Callosciurus, Rodentia: Sciuridae) from the Callosciurus erythraeus/finlaysonii complex and their distribution in eastern Indochina. Raffles Bulletin of Zoology, 67, 459489. https://doi.org/10.26107/RBZ-2019-0037

Chapter 4 Visualization and annotation of phylogenetic trees: ggtree. https://guangchuangyu.github.io/ggtree-book/chapter-ggtree.html. Accessed October 28th 2022.

Gabrielli, M. et al. (2014). Genetic Characterization of Callosciurus (Rodentia: Sciuridae) Asiatic Squirrels Introduced in Argentina. Italian Journal of Zoology 81(3), 328–343. http://doi.org/10.1080/11250003.2014.940006

Jackson, S. M., Li, Q., Wan, T., Li, X.Y., Yu, F.H., Gao, G., He, L.K., Helgen, K. M., & Jiang, X.L. (2022). Across the great divide: Revision of the genus Eupetaurus (Sciuridae: Pteromyini), the woolly flying squirrels of the Himalayan region, with the description of two new species. Zoological Journal of the Linnean Society, 194(2), 502–526. https://doi.org/10.1093/zoolinnean/zlab018

Li, G., Lwin, Y. H., Yang, B., Qin, T., Phothisath, P., Maung, K.W., Quan, R.C., & Li, S. (2020). Taxonomic revision and phylogenetic position of the flying squirrel genus Biswamoyopterus (Mammalia, Rodentia, Sciuridae, Pteromyini) on the northern Indo-China peninsula. ZooKeys, 939, 65–85. https://doi.org/10.3897/zookeys.939.31764

Menéndez, I., Gómez Cano, A. R., Cantalapiedra, J. L., Peláez-Campomanes, P., Álvarez-Sierra, M. Á., & Hernández Fernández, M. (2021). A multi-layered approach to the diversification of squirrels. Mammal Review, 51(1), 66–81. https://doi.org/10.1111/mam.12215

Model choice exercise. https://gtpb.github.io/MEVR16/ml/example1.html. Accessed October 28th 2022.

Moritz, C. (1995). Uses of molecular phylogenies for conservation. Proceedings of the Royal Society B: Biological Sciences, 349(1327). https://doi.org/10.1098/rstb.1995.0097

Popadin, K., Gunbin, K., Peshkin, L., Annis, S., Fleischmann, Z., Kraytsberg, G., Markuzon, N., Ackermann, R. R., & Khrapko, K. (2017). Mitochondrial pseudogenes suggest repeated inter-species hybridization among direct human ancestors. Evolutionary Biology. https://doi.org/10.1101/134502

Sequence reversing and complementing. https://stuff.mit.edu/afs/athena/software/r/current/lib/R/library/Biostrings/html/reverseComplement.html. Accessed October 28th 2022.

Shaw, K. L. (2002). Conflict between nuclear and mitochondrial DNA phylogenies of a recent species radiation: What mtDNA reveals and conceals about modes of speciation in Hawaiian crickets. Proceedings of the National Academy of Sciences, 99(25), 16122–16127. https://doi.org/10.1073/pnas.242585899

Suárez-Díaz, E., & Muñoz, V. H. (2008). History, objectivity, and the construction of molecular phylogenies. Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences, 39(4), 451–468. https://doi.org/10.1016/j.shpsc.2008.09.002

Two Histograms with melt colors. https://r-graph-gallery.com/2-two-histograms-with-melt-colors.html. Accessed October 28th 2022.

Visualizing and Annotating Phylogenetic Trees with R+ggtree. https://4va.github.io/biodatasci/r-ggtree.html. Accessed October 28th 2022.

Writing reproducible reports knitr with R Markdown. https://kbroman.org/Tools4RR/assets/lectures/03_knitr_Rmd.pdf. Accessed October 28th 2022.

Yu, G., Smith, D. K., Zhu, H., Guan, Y., & Lam, T. T. (2017). ggtree: An r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. Methods in Ecology and Evolution, 8(1), 28–36. https://doi.org/10.1111/2041-210X.12628